Ranking Automation: User Guide

Version 1.0

Welcome to your new Ranking Automation tool! This guide will walk you through everything you need to know, from the initial technical setup to strategic daily use, all in simple, easy-to-understand language.

This powerful tool automates the process of checking Google search rankings for your website, saving you hours of manual work and providing accurate, up-to-date data directly in a Google Sheet.

Sheet Link: Ranking Automation

Table of Contents

Chapter 4: Strategies for Effective & Stealthy Ranking Checks

- Managing Keywords for Different Locations
- How to Handle CAPTCHAs
- When to Refresh Your Master Profile
- Customizing Delays for Stealth

Chapter 5: How the Magic Happens (A Simple Look at the Script)

- What are Python and Selenium?
- The Journey of a Keyword: The Script's Logic
- Important Settings to Know

Appendix: Troubleshooting & Frequently Asked Questions (FAQ)

- Common Problems and Solutions

Chapter 1: Introduction to the Tool

What is this Automation Tool?

Imagine you have a list of 100 keywords and you need to know where your website ranks for each one on Google. Normally, you would have to search each keyword, scroll through pages of results, and manually record the position. This is incredibly time-consuming and prone to errors.

This tool solves that problem.

It is a smart system that uses a Python script to automatically perform Google searches for a list of keywords you provide in a Google Sheet. It mimics human behavior to avoid being blocked, finds your website's rank for each keyword, and writes the result back into the same Google Sheet. It's your personal SEO assistant for tracking performance.

Who is this Tool For?

This tool is designed for anyone who needs to track their website's search engine ranking without spending hours doing it manually. It's perfect for:

- **SEO Specialists:** To monitor keyword performance for clients or in-house projects.
- **Digital Marketers:** To track the impact of marketing campaigns on search visibility.
- **Business Owners:** To keep an eye on their website's position for important search terms.
- **Content Creators:** To see how their articles and blog posts are ranking on Google.

Key Features at a Glance

- **Fully Automated Ranking Checks:** Just add your keywords and target URL to a Google Sheet, and the tool does the rest.
- **Google Sheets Integration:** Manages all input and output in one convenient, cloud-based Google Sheet.
- **Human-Like Browsing:** Uses a real Chrome browser profile, random delays, and rotating user agents to appear like a real user, reducing the chance of being blocked.
- **CAPTCHA Handling:** If a CAPTCHA is detected, the script pauses, sends you an email alert, and waits for you to solve it manually before continuing.
- **Detailed Logging:** Every action—each search, page scraped, and result found—is recorded in a local log file so you always know what's happening.
- **Email Error Alerts:** If the script crashes or requires your attention (like for a CAPTCHA), it automatically sends an email to you.

How It Works: A Simple Overview

The process is straightforward. Think of it as giving a set of instructions to a very efficient robot.

1. **You Provide the List:** You add keywords and your website URL to the 'Ranking Automator' sheet.
2. **The Automation Wakes Up:** You run the script from your computer.
3. **It Reads the First Keyword:** The script opens the Google Sheet and reads the first keyword and your target URL.
4. **It Opens a Browser:** It launches a Chrome browser, just like you would.
5. **It Searches Google:** It navigates to Google, types the keyword into the search bar, and hits Enter.
6. **It Scans the Results:** The script carefully reads the search results on the first page, looking for your website's URL. If it doesn't find it, it clicks to the next page and continues searching (up to 10 pages).
7. **It Records the Rank:** Once it finds your URL, it records the position (e.g., 5, 12, 25) and the exact URL it found. If it doesn't find it within 10 pages, it records "Not Found".
8. **It Updates the Sheet:** The script writes the rank and the found URL back into the correct row in your Google Sheet.
9. **It Repeats:** It moves to the next keyword in your list and starts the entire process over again until all keywords are checked.

Chapter 2: Getting Started: The One-Time Setup Process

This chapter will guide you through the one-time setup of your automation tool. Follow these steps carefully to ensure everything works perfectly.

Prerequisites: What You'll Need

1. **Python:** Ensure you have Python installed on your computer (Version 3.8 or newer is recommended).
2. **A Code Editor:** A program like Visual Studio Code is highly recommended for editing the configuration files.
3. **A Google Account:** A dedicated Gmail account (e.g., `yourcompany.automation@gmail.com`) is recommended for this process.
4. **Google Chrome:** The automation uses Chrome, so you must have it installed.

Step 1: Setting Up Your Google Cloud Platform (GCP) API

This allows the script to securely access your Google Sheet.

1. Go to the Google Cloud Console.
2. Create a new project. Give it a name like "Ranking Automation Project".
3. In the search bar at the top, search for and **enable** these two APIs:
    - **Google Drive API**
    - **Google Sheets API**
4. Go to "Credentials" from the left-hand menu (or search for it).
5. Click **+ CREATE CREDENTIALS** and select **Service account**.
6. Give the service account a name (e.g., "sheets-automator") and click **CREATE AND CONTINUE**.
7. For "Role", select **Project** > **Editor**. Click **CONTINUE**, then **DONE**.
8. You will now see your new service account in the list. Click on its email address.
9. Go to the **KEYS** tab. Click **ADD KEY** > **Create new key**.
10. Choose **JSON** as the key type and click **CREATE**.
11. A `.json` file will be downloaded to your computer. **This file is your password! Keep it safe.** Rename it to `gcp_credentials.json`.

Step 2: Setting Up Your Google Sheet

This is where you will manage your keywords and see the results.

1. **Make a Copy:** Open this Google Sheet template link and make your own copy: Ranking Automation Sheet Template
    - Click **File > Make a copy**. Save it to your own Google Drive.
2. **Share the Sheet:**
    - Open the `gcp_credentials.json` file you downloaded. Find the line that says `"client_email"`.
    - Copy the email address (it looks like `something@...gserviceaccount.com`).
    - In your new Google Sheet, click the **Share** button.
    - Paste the service account's email address and give it **Editor** permissions. Click **Send**.
3. **Understanding the Columns:**
    - **Keyword:** The search term you want to check.
    - **Company1:** The base URL of the website you are tracking (e.g., `https://www.mywebsite.com/`).
    - **Rankings:** The script will automatically fill this column with the rank number.
    - **Ranking URL:** The script will fill this with the exact URL it found in the search results.
4. **Populate with Data:** You can now add your keywords and target URL. We have also provided a sample CSV file (`Ranking Automation - Ranking Automator.csv`) that you can import to see the correct format.

Step 3: Setting Up Your Local Project Folder & Python Environment

This organizes your files and keeps the project's dependencies separate.

1. **Create a Project Folder:** On your computer, create a new folder named `Ranking_Automation`.
2. **Move Project Files:** Place all the provided project files (`.py` files, `requirements.txt`, etc.) and your `gcp_credentials.json` file into this folder.
3. **Open a Terminal:**
    - On Windows, open Command Prompt or PowerShell.
    - On Mac/Linux, open Terminal.

4. **Navigate to Your Folder:** Use the `cd` command to navigate into your `Ranking_Automation` folder.
    - Example: `cd C:\Users\YourName\Documents\Ranking_Automation`
5. **Create a Virtual Environment:** This creates an isolated Python "bubble" for this project.
    - Run this command: `python -m venv venv`
6. **Activate the Environment:**
    - On Windows: `venv\Scripts\activate`
    - On Mac/Linux: `source venv/bin/activate`
    - You will see `(venv)` appear at the start of your terminal line.
7. **Install Required Packages:**
    - Run this command: `pip install -r requirements.txt`
    - This will automatically install Selenium, gspread, and all other necessary libraries.

Step 4: Configuring the Automation (

This file contains all the important settings for the tool. Open `config.py` in your code editor.

1. **`PROJECT_ROOT`**: Update this path to the absolute path of your `Ranking_Automation` folder.
    - Example: `r"C:\Users\YourName\Documents\Ranking_Automation"`
2. **`CHROME_PROFILE_PATH`**: This should be correct by default, as it points inside your project root. No changes are needed unless you want to store the profile elsewhere.
3. **`SHEET_NAME`**: Enter the exact name of your Google Sheet copy (e.g., "Copy of Ranking Automation").
4. **`WORKSHEET_NAME`**: Enter the name of the tab within your sheet. By default, it is "Ranking Automator".
5. **Email Notification Settings (**:
    - **`ENABLE_EMAIL_NOTIFICATIONS`**: Set to `True` to receive alerts.
    - **`SENDER_EMAIL`**: Enter the Gmail address you want to send alerts *from*.
    - **`SENDER_PASSWORD`**: This is **NOT** your regular Gmail password. It's an "App Password".
        - Go to your Google Account settings > Security.
        - Turn on 2-Step Verification.
        - Go to "App passwords", create a new one for "Mail" on "Windows Computer", and copy the 16-digit password it gives you. Paste it here.
    -

- **`RECIPIENT_EMAIL`**: Enter the email address(es) where you want to receive alerts.

Step 5: Creating Your Master Chrome Profile

This step creates a dedicated, logged-in Chrome profile for the automation to use, which helps avoid CAPTCHAs.

1. Make sure your virtual environment is still active in your terminal.
2. Run the profile creation script: `python create_master_profile.py`
3. A new Chrome window will open. **You have 90 seconds to perform the following actions:**
    - Go to `google.com`.
    - Sign in to your dedicated Google Account.
    - Accept any cookie pop-ups.
    - **IMPORTANT:** If Google asks to "Turn on sync?", click **"No thanks"**.
    - Once you are signed in, you can manually close the browser.
4. The script will finish, and your profile is now ready.

Step 6: Activating the Automation

Congratulations! Your Ranking Automation tool is now fully set up and ready to go.

To run the tool, simply open your terminal, navigate to your project folder, activate the virtual environment, and run the main script.

1. Navigate to your folder: `cd path\to\Ranking_Automation`
2. Activate venv: `venv\Scripts\activate`
3. Run the script: `python ranking_automator.py`

The script will now begin its work. You can watch its progress in the terminal and see the results appear in your Google Sheet in real-time.

Chapter 3: Using the Automation Day-to-Day

Now that the setup is complete, using the tool is simple.

Your Daily Workflow

1. **Add New Keywords:** Open your Google Sheet. Go to the 'Ranking Automator' tab and add new keywords and the corresponding target URL to the bottom of the list.
2. **Leave Ranking Columns Blank:** Do not enter anything in the `Rankings` or `Ranking URL` columns for new keywords. The script uses the empty cells to know which keywords to process.
3. **Run the Automation:** Follow the steps in "Step 6: Activating the Automation" from the previous chapter to start the process.
4. **Check Your Progress:**
    - Watch the terminal for live updates on which keyword is being processed.
    - Open your Google Sheet to see the `Rankings` and `Ranking URL` columns being filled in as the script works.

Running the Automation

To start the tool at any time, follow these three commands in your terminal:

```
# 1. Go to your project folder
cd C:\Path\To\Your\Ranking_Automation

# 2. Activate the Python environment
venv\Scripts\activate

# 3. Run the main script
python ranking_automator.py
```

To stop the script while it's running, simply press **Ctrl + C** in the terminal window.

Understanding the Output

- **A number (e.g.,** This is the rank of your website for that keyword.
- **:** This means the script scanned the first 10 pages of Google results and did not find your URL.
- **Ranking URL:** This column shows the exact page from your website that is ranking, which is useful for confirming the correct page is showing up.

Monitoring the Log File

For a detailed, behind-the-scenes look, you can open the `ranking_automator.log` file in your project folder. This file records every single step the script takes, including:

- Which keyword it's starting.
- Which page of the search results it's scraping.
- When it finds a result.
- Any warnings or errors it encounters.

Chapter 4: Strategies for Effective & Stealthy Ranking Checks

This tool is powerful, but using it strategically will give you the best and most reliable results.

Managing Keywords for Different Locations

If you need to check rankings from different geographical locations (e.g., 100 keywords for Mumbai and 100 for Bangalore), you cannot do this directly in the script. However, you can use a browser extension to achieve this.

The Strategy:

The script launches a fresh browser instance for each keyword. This means a location-changing extension (like a VPN) can set a new location before each search begins.

How to Implement:

1. **Install a Location Changer Extension:** Install a VPN or location-spoofing extension in your regular Chrome browser.
2. **Enable in Master Profile:** When you run `create_master_profile.py`, make sure to also install and configure this extension in the special Chrome window that opens. Set it to your first desired location (e.g., Mumbai).
3. **Structure Your Keyword List:** In your Google Sheet, group all keywords for one location together. For example, put all 100 Mumbai keywords first.
4. **The "Buffer" Technique:** To switch locations, you need to pause the script. The easiest way is to plan for it.
    - After your 100 Mumbai keywords, add 5-10 "dummy" or unimportant keywords.
    - Run the script. It will process the 100 Mumbai keywords.
    - When it starts on the dummy keywords, press **Ctrl + C** to stop the script.
    - Manually open your Chrome browser, switch your location-changer extension to the next location (e.g., Bangalore).
    - Relaunch the script (`python ranking_automator.py`). It will pick up where it left off, but now all subsequent searches will be from the new location.

How to Handle CAPTCHAs

Even with human-like behavior, Google may occasionally present a "I'm not a robot" CAPTCHA.

1. **The Script Pauses:** The tool will detect the CAPTCHA and immediately pause its execution.
2. **You Get an Alert:** If you configured email notifications, you will receive an email telling you that manual action is required. The terminal will also display a large message.
3. **Solve it Manually:** The Chrome browser window used by the script will remain open on your screen. Simply go to that window and solve the CAPTCHA yourself.
4. **The Script Resumes:** Once you solve it, the script will automatically detect that the CAPTCHA is gone and will resume its work from where it left off.

When to Refresh Your Master Profile

Over time, the login session in your Chrome profile might expire. If you notice the script is consistently getting CAPTCHAs or failing to load Google correctly, it's time for a refresh.

1. Make sure your virtual environment is active.
2. Run the refresh script: `python refresh_profile.py`
3. This will delete the old profile and guide you through the same login process as the initial setup. This gives the automation a fresh, logged-in session to work with.

Customizing Delays for Stealth

In the `config.py` file, the `DELAY_CONFIG` dictionary controls how long the script waits between actions. The default values are very small for speed. If you are checking a very large number of keywords and are concerned about being blocked, you can increase these values to make the script behave even more like a human.

Example: Change `"serp_read": {"min": 0.01, "max": 0.01}` to `"serp_read": {"min": 2, "max": 5}` to make the script pause for a random time between 2 and 5 seconds on each search results page, as if a person were reading it.

Chapter 5: How the Magic Happens (A Simple Look at the Script)

You don't need to understand code to use this tool, but it can be helpful to know what's happening behind the scenes.

What are Python and Selenium?

- **Python:** A popular programming language that is excellent for automation. The scripts you are running are written in Python.
- **Selenium:** A powerful tool that allows Python to control a web browser. It's the "magic" that lets the script open Chrome, type in the search bar, and click buttons.

The Journey of a Keyword: The Script's Logic

When you run the script, it follows a very logical, step-by-step process for each keyword in your list:

1. **Wake Up & Connect:** The script starts and connects to the Google Sheets API using your credentials.
2. **Read the List:** It downloads the entire list of keywords and URLs from your specified worksheet.
3. **Launch Browser:** It opens a new Chrome window using the special "Master Profile" you created, which is already logged into Google.
4. **Pick a Keyword:** It takes the first keyword from the list that doesn't have a rank yet.
5. **Go to Google:** It navigates the browser to `google.com`.
6. **Perform Search:** It finds the search box, types the keyword in a human-like way (character by character), and presses Enter.
7. **Analyze Page 1:** It waits for the results to load and then carefully scans each organic result, checking if the URL matches your target URL.
8. **Found or Not Found?**
    - **If Found:** It records the rank and the exact URL, writes this data to the Google Sheet, and moves to the next keyword.
    - **If Not Found:** It looks for the "Next" page button.
9. **Go to Next Page:** If the "Next" button exists and it hasn't checked 10 pages yet, it clicks the button and repeats the analysis on the new page.
10. **End of Search:** If it reaches the end of the results or finishes page 10 without finding your URL, it marks the rank as "Not Found" in the sheet.
11. **Repeat:** The script takes a short break and then starts the entire process over with the next keyword in the list.

12. **Shutdown:** Once all keywords are processed, it closes the browser and the script finishes.

Important Settings to Know?

These are defined in `config.py` and control key behaviors:

- **`MAX_PAGES_TO_SCRAPE`**: Set to `10` by default. This tells the script to stop searching for a keyword after the 10th page of results.
- **`CAPTCHA_WAIT_TIMEOUT`**: Set to `900` seconds (15 minutes). This is the maximum amount of time the script will wait for you to solve a CAPTCHA before it gives up on that keyword and moves to the next one.

How to use for different locations?

This can be done using GS locator extension what you have to do:

- Add 5 random keywords above the keywords you want rankings of, While the script is running for these 5 keywords change the location in extension.
- The extension works after the reload so, it will work the moment it is changed and the next keyword reloads the page.

How to use it for perfect ranking?

To run for perfect ranking run incognito_main.py.

Note: You can use this project for exact url + domain base search. The scripts are searching for the provided url. You can add this exact url or else just domain in the company column of the sheet. (eg:. https://_____.com/).

Appendix: Troubleshooting & Frequently Asked Questions (FAQ)

Q: I ran the script, but it crashed immediately with a "FileNotFoundError".
A: This is almost always an incorrect path in your `config.py` file.

- **Check** : Make sure this is the full, correct path to your project folder. Use absolute paths (e.g., `C:\...` or `/Users/...`), not relative ones.
- **Check** : Ensure the `gcp_credentials.json` file is in the main project folder and its name is spelled exactly right.

Q: The script runs, but I get an error about "gspread.exceptions.SpreadsheetNotFound".
A: This means the script can't find your Google Sheet.

- Check the `SHEET_NAME` in `config.py`. It must match the name of your Google Sheet *exactly*, including spaces and capitalization.
- Make sure you have shared the Google Sheet with the `client_email` from your `gcp_credentials.json` file and given it "Editor" permissions.

Q: The script says "CAPTCHA DETECTED" but I don't see a browser window.
A: The browser window may be hidden behind other windows. Look for a new Chrome icon in your taskbar and click on it to bring the window to the front so you can solve the CAPTCHA.

Q: All my rankings are coming back as "Not Found", but I know my site ranks!
A: There are a few possible causes:

- **URL Mismatch:** Check the `Company1` URL in your sheet. It must be a part of the URL that appears in Google's search results. For example, if your site is `https://www.mywebsite.com`, using `mywebsite.com` is a safe bet. Avoid using `http://` if your site is `https://`.
- **Location Mismatch:** The script is searching from the location of your computer/server. If you are tracking ranks for a different city or country, the results will be different. See Chapter 4 for the strategy on handling different locations.
- **Google HTML Changes:** Google sometimes changes the layout of its search results page. If this happens, the `serp_selectors.py` file may need to be updated by a developer.

Q: How do I completely stop the automation?

A: Go to the terminal window where it is running and press **Ctrl + C**. It will not run again until you manually start it.